A music structure inference algorithm based on symbolic data analysis

Gabriel Sargent, Stanislaw Andrzej Raczynski, Frédéric Bimbot, Emmanuel Vincent, Shigeki Sagayama

A MUSIC STRUCTURE INFERENCE ALGORITHM BASED ON SYMBOLIC DATA ANALYSIS

Gabriel SARGENT
Université de Rennes 1
IRISA (UMR 6074)
gabriel.sargent@irisa.fr

Stanisław A. RACZYŃSKI
Graduate School of Information, Science and Technology
The University of Tokyo
raczynski@hil.t.u-tokyo.ac.jp

Frédéric BIMBOT
CNRS
IRISA (UMR 6074)
frederic.bimbot@irisa.fr

Emmanuel VINCENT
INRIA
Centre INRIA Rennes
Bretagne Atlantique
emmanuel.vincent@inria.fr

Shigeki SAGAYAMA
Graduate School of Information, Science and Technology
The University of Tokyo
sagayama@hil.t.u-tokyo.ac.jp

ABSTRACT

The present document describes a music structure inference algorithm submitted to the MIREX 2011 evaluation campaign (structural segmentation task). It consists of 3 stages : symbolic feature extraction, structural segment boundary estimation, and structural segment clustering. We consider as inputs chord estimations from the system of Ueda *et al.* in [5], expressed at the 2-beat scale. Beats and downbeats are estimated by the system of Davies *et al.* [2, 3]. The structural segmentation step uses a regularity-constrained Viterbi approach. It assumes that the structure of pop songs is generally based on a few typical segments, whose sizes are called structural pulsation periods [1]. The segments are then clustered according to their similarity, through the minimization of an adaptive model selection criterion.

1. FEATURE EXTRACTION

The considered music piece is transformed into a sequence of estimated chord symbols thanks to the system of Ueda *et al.* [5]. It considers 7 chord types for each of the 12 semitones : minor, major, augmented, diminished and seventh (plus the "no chord" type).

The downbeat estimator by Davies *et al.* [2] is tuned so as to consider 4 beats per bar. For more information about the beat tracker used, please refer to the description of the non-causal system in [3].

The estimations of beats and downbeats allow to build the beat scale whose beat period is closer to 1 second, and synchronous to the downbeat scale. The temporal units obtained are referred as "snaps" and used for structural analysis in [1].

We associate to each snap the symbol of the chord which appears the most in the time frame limited by this snap and the next one. Then, the set of features used by the structure inference system is a sequence of chords symbols expressed at the snap scale.

2. MUSIC STRUCTURE INFERENCE SYSTEM

2.1 Viterbi-based structural segmentation under regularity constraints

This system seeks for the best structural segmentation through a cost optimization process. Let $S = \{s_k\}_{1 \leq k \leq n}$ be a segmentation of the sequence of features X describing a music piece. We assume that the cost function can be written :

$$C(S) = \sum_{k=1}^{n} \Gamma(s_k) \tag{1}$$

with

$$\Gamma(s_k) = \Phi(s_k) + \lambda \Psi(s_k) \tag{2}$$

- $\Phi(s_k)$ is a data-based segmentation cost which assign a low value to segments made of sequences of features repeated elsewhere in the song.
- $\Psi(s_k)$ is a regularity cost which takes a low value when the segment size is close to a particular value τ.
- λ is a balance parameter between these two costs.

The minimization of this cost is achieved by means of a Viterbi algorithm. This system corresponds to "System2" described in [4], with the following regularity cost :

$$\Psi(s_k) = \sqrt{\frac{m_k}{\tau} - 1} \tag{3}$$

m_k is the length of segment s_k.

The typical segment size is set to $\tau = 16$ snaps, and $\lambda = 0.15$ according to a former experiment on the RWC popular database.

2.2 Structural segment clustering through automaton modeling

We consider that the set of features of a music piece can be interpreted as a sequence of states of an automaton. Segments modeled by the same branch are considered as a cluster, *i.e.* they typically correspond to similar sequences of symbols. The probability of the observed song is therefore modeled as the product of two types of transition probabilities: transitions between the automaton branches (between two successive segments) and between the automaton states (between two successive symbols in a segment).

As a lot of automata can model a single music piece, let's precise the automaton space we consider and the selection criterion used.

2.2.1 Automaton space :

We consider a set of automata with different number of branches. A first automaton is built by associating one branch to each structural segment. Additional automata are recursively built by fusing pairs of branches into a single branch, until only a single branch is left.

Here, the order of fusion of the branches is set according to their similarity : considering the automaton with i branches, the automaton with $(i-1)$ branches is obtained by fusing the two branches with lowest distance.

The distance between the two branches, which model respectively two sets (A and B) of structural segments, is the minimum of the edit distance between each segment of A and each segment of B (i.e. the number of symbol addition, deletion and substitution errors) [1] .

2.2.2 Adaptive criterion for automaton selection :

To each automaton is associated its probability to produce the sequence of symbols describing the whole song. Let i be the number of automaton branches and P_i this probability, we have :

$$P_i = P_{B_i} P_{S_i} \tag{4}$$

P_{B_i} is the probability of the observed sequence of segment clusters (branches),

P_{S_i} is the probability of the observed sequence of symbols given the clustering of segments.

In information theory, searching for the automaton with highest P_i is equivalent to searching for the automaton with the lowest quantity of information $-\log(P_i)$. As we observe in pop music pieces that the number of labels rarely reaches one, or the number of segments, we add a penalty term y that is affine in the number of branches and that gives the same quantity of information to the smallest and largest automata (with respectively one branch, and n branches if the song is made of n segments).

We then search for the automaton which minimizes the following criterion :

$$y - \log(P) = \{y_i - \log(P_i)\}_{1 \leq i \leq n} \tag{5}$$

where

$$y_i = \frac{\log(P_n) - \log(P_1)}{n-1}(i-1) + \log(P_1) \tag{6}$$

Former experiments showed that the performances obtained with this criterion were close to the ones obtained with BIC or AIC criteria.

Each cluster is then associated to a segment label (A, B, C...) which is returned with the segment borders by the structure inference system.

3. ACKNOWLEDGEMENTS

The authors would like to thank Yushi Ueda and Nobutaka Ono for their help in the collection of chord transcriptions used in this article. This work was partly supported by the Quaero project [2] funded by Oseo and by the associate team VERSAMUS [3] funded by INRIA.

4. REFERENCES

[1] F. Bimbot, O. Le Blouch, G. Sargent and E. Vincent, " Decomposition into autonomous and comparable blocks : a structural description of music pieces", *Proceedings of the International Symposium on Music Information Retrieval*, pp. 189–194, 2010.

[2] M. E. P. Davies, "Towards Automatic Rhythmic Accompaniment" (Chapter 6), Ph.D. Thesis, Department of Electronic Engineering, Queen Mary University of London, 2007

[3] A. M. Stark, M. E. P. Davies and M. D. Plumbley, "Real-Time Beat-Synchronous Analysis of Musical Audio" *Proceedings of the 12th Int. Conference on Digital Audio Effects*, Como, Italy, pp. 299–304, September 1-4, 2009.

[4] G. Sargent, F. Bimbot and E. Vincent, "A regularity-constrained Viterbi algorithm and its application to the structural segmentation of songs" to appear in *Proceedings of the International Symposium on Music Information Retrieval*, 2011.

[5] Y. Ueda, Y. Uchiyama, T. Nishimoto, N. Ono and S. Sagayama, "HMM-based Approach for Automatic Chord Detection Using Refined Acoustic Features", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 5506–5509, March 2010.

[1] we currently use the edit distance script by Miguel Castro, available at http://www.mathworks.com/matlabcentral/fileexchange/213-editdist-m

[2] http://www.quaero.org/

[3] http://versamus.inria.fr/